

MULTIMÍDIA

Trabalho 5 – Filtros e Compressão de imagens

Aluno: Hugo Bourguignon Rangel

IPRJ - Nova Friburgo - Rio de janeiro

Sumário

Filtros.....	3
Introdução.....	3
Filtros Passa-Baixa	3
Filtros Passa-Alta	4
Exemplos Práticos (Scilab).....	4
Compressão de imagens	11
Introdução.....	11
LZW (Lempel-ZivWelch)	12
Compactação JPEG (transformada DCT+Huffman)	14
Exemplos Práticos de compressão (Scilab)	15
Referencias.....	21

Filtros

Introdução

Os filtros são, basicamente, aplicações de técnicas de transformação (utilizando operadores e mascaras) em imagens, com objetivo de corrigir, suavizar ou realçar determinadas características da imagem.

A imagem pode ser descrita como um amontoado de pixels (matriz de pixels). Logo, quando se usa um filtro, a filtragem ocorre de pixel em pixel; alterando as características de cada pixel de acordo com as características dos pixels na vizinhança do próprio. Geralmente a característica principal desses pixels é determinada como “nível de cinza”, sendo ela base para todas as transformações de filtragem.

Os filtros se dividem em dois grupos, sendo eles os filtros lineares e os filtros não-lineares:

- **Filtros lineares:** Suavizam, realçam detalhes e minimizam ruídos da imagem sem alterar o nível médio de cinza da imagem (mantendo o mesmo nível médio de cinza da imagem original).
- **Filtros não-lineares:** Suavizam, realçam detalhes e minimizam ruídos da imagem sem o compromisso de manter o nível médio de cinza da imagem (nível médio de cinza diferente da imagem original).

Os filtros lineares e não lineares podem ser implementados utilizando o **domínio espacial** (processos que atuam diretamente sobre os pixels da imagem) ou o **domínio da frequência** (processos que atuam sobre a transformada de *Fourier* da imagem de entrada).

Filtros Passa-Baixa

Se tratam de filtros lineares que possuem o objetivo de suavizar e atenuar os contrastes entre detalhes de uma imagem.

Considerando o domínio da frequência, um filtro passa-baixa suaviza lugares com altas frequências, pois esses lugares na imagem caracterizam transições abruptas no nível de cinza dos pixels nessa região. O efeito visual que corresponde a um filtro passa-baixa se chama *IsMOOTHING* (suavização).

A suavização dos detalhes de uma imagem tende a diminuir o efeito do ruído sobre a imagem, contudo, acaba diminuindo a nitidez e definição da imagem; em alguns casos, provocando um barramento na imagem.

Filtros Passa-Alta

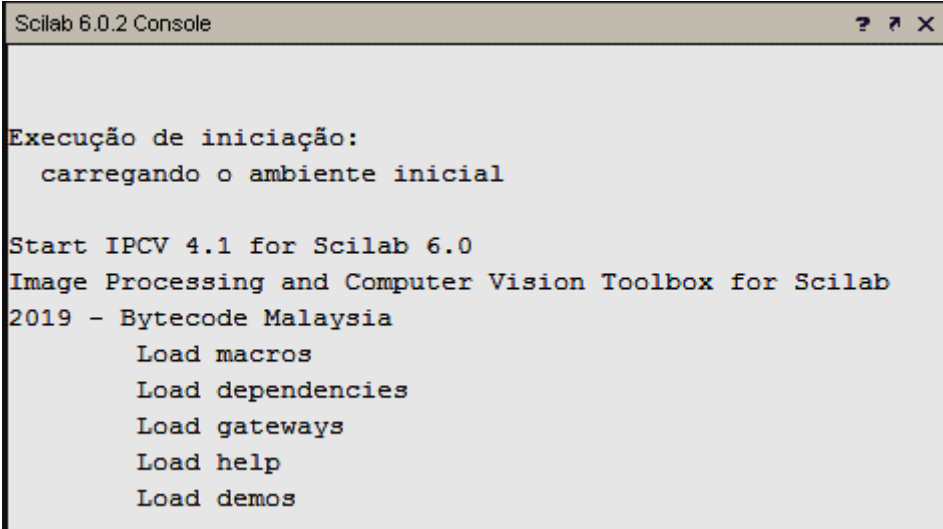
Se tratam de filtros que tornam os detalhes de uma imagem mais nítidos, através do realce do contraste entre regiões distintas (apelidadas de bordas de contraste)

Considerando o domínio da frequência , um filtro passa-alta atenua ou elimina as baixas frequências (onde se encontram os lugares da imagem com menos contraste de níveis de cinza) , assim , realçando as altas frequências (lugares com altos contrastes de níveis de cinza).O efeito visual que corresponde a um filtro passa-alta se chama *Sharpening* (“agudização” - tornar algo mais perceptível -).

Como já mencionado o filtro passa-alta deixa os detalhes da imagem mais nítidos, facilitando a percepção dos detalhes e aumentando a definição; porém, acaba enfatizando os ruídos da imagem (deixando os ruídos mais evidentes nas imagens).

Exemplos Práticos (Scilab)

Através do open source software de computação numérica **Scilab** e seu respectivo modulo de processamento de imagem e visão computacional **IPCV** (Scilab Image Processing & Computer Vision) foi possível aplicar na pratica os filtros passa-alta e passa-baixa. Abaixo é exibido a iniciação do console do Scilab (na versão 6.0.2) com o modulo IPVc carregado no ambiente de trabalho.



```
Scilab 6.0.2 Console

Execução de iniciação:
  carregando o ambiente inicial

Start IPCV 4.1 for Scilab 6.0
Image Processing and Computer Vision Toolbox for Scilab
2019 - Bytecode Malaysia
  Load macros
  Load dependencies
  Load gateways
  Load help
  Load demos
```

Agora, serão apresentados os códigos, resultados e componentes das praticas de criação de um filtro passa-alta e um filtro passa-baixa (ambos bem simples).

- **Filtro passa-baixa**

Com o ambiente de trabalho já pronto foi escolhido um filtro de média ponderada 3X3 para ser aplicado. Abaixo segue a matriz que representa a máscara do filtro.

$$\frac{1}{10} * \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Com o filtro já selecionado, foi necessário ler a imagem original (de própria autoria) no Scilab. A seguir o código para fazer o tal.

```

97 //-----/imagem original/-----//
98
99 subplot(221)
100 img_original = imread('C:\Users\Arma-X\Desktop\Muitimidia\Trabalho-5\Imagens_originais\teste-
.tif')
101 imshow(img_original);
102 title('Imagem original');

```

Com a imagem original já carregada agora é possível aplicar o filtro; abaixo segue o código que corresponde a parte de aplicação do filtro (e sua máscara *matriz*).

```

111 //-----/Filtro passa-baixa/-----//
112
113 subplot(223)
114 a=double(img_original);
115 [m,n] = size(a);
116 w = [1 1 1; 1 2 1; 1 1 1];
117 for i=2:m-1
118     for j=2:n-1
119         b(i,j) = (w(1)*a(i-1,j+1) + w(2)*a(i,j+1) + w(3)*a(i+1,j+1) + w(4)*a(i-1,j) + w(5)*a(i,j) + w(6)*
a(i+1,j) + w(7)*a(i-1,j-1) + w(8)*a(i,j-1) + w(9)*a(i+1,j-1))/10;
120     end
121 end
122 c=uint8(b);
123 imshow(c);
124 imwrite(c, 'C:\Users\Arma-X\Desktop\Muitimidia\Trabalho-5\Imagens_com_filtro\teste(pass-baixa)
.tif')
125 title('Imagem com o filtro passa-baixa aplicado');

```

- **Filtro passa-alta**

Com o ambiente de trabalho já pronto foi escolhido um filtro 3X3, e sua máscara contendo coeficientes positivos no centro e negativos em suas extremidades (característica dos filtros passa-alta). para ser aplicado. Abaixo segue a matriz que representa a máscara do filtro.

-1	-1	-1
-1	8	-1
-1	-1	-1

Com o filtro já selecionado e a imagem já carregada no processo do filtro passa-baixa. Foi possível aplicar o filtro; abaixo segue o código que corresponde a parte de aplicação do filtro (e sua máscara *matriz*).

```

127 //-----/Filtro passa-alta/-----//
128
129 subplot(224)
130 z = [-1 -1 -1; -1 8 -1; -1 -1 -1];
131 for i=2:m-1
132     for j=2:n-1
133         d(i,j) = (z(1)*a(i-1,j+1) + z(2)*a(i,j+1) + z(3)*a(i+1,j+1) + z(4)*a(i-1,j) + z(5)*a(i,j) + z(6)*a(i+1,j) + z(7)*a(i-1,j-1) + z(8)*a(i,j-1) + z(9)*a(i+1,j-1))/9;
134     end
135 end
136 e = uint8(d);
137 imshow(e);
138 imwrite(e, 'C:\Users\Arma-X\Desktop\Muitimidia\Trabalho-5\Imagens_com_filtro\teste(pass-alta).tif');
139 title('Imagem com o filtro passa-baixa aplicado');

```

Com os códigos já prontos para ambos os filtros agora é possível gerar os resultados para cada um, e como complemento para formar quatro resultados em uma mesma janela gráfica foi aplicado o negativo da imagem original. A seguir é apresentado o código de aplicação do negativo da imagem.

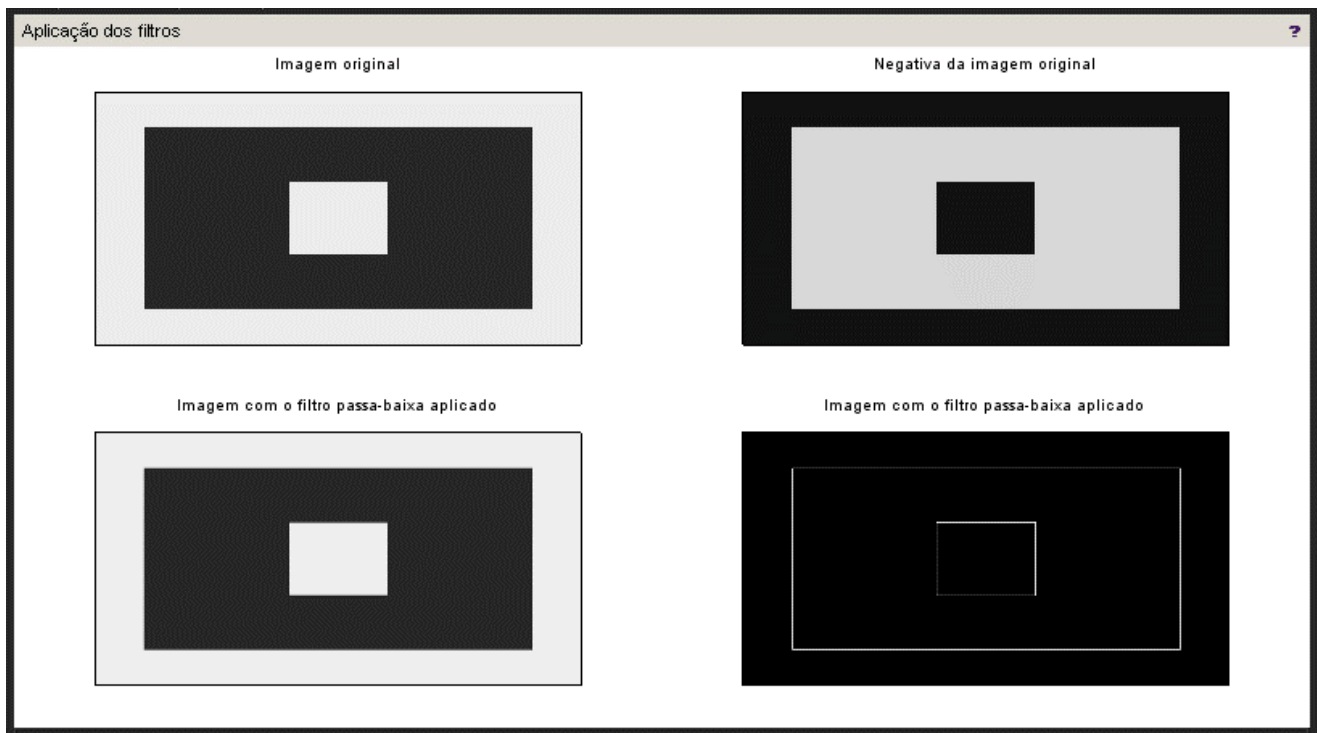
```

104 //-----/Negativa da imagem original/-----//
105
106 subplot(222)
107 img_negativa_ = imcomplement(img_original);
108 imshow(img_negativa_);
109 title('Negativa da imagem original');

```

• Resultados

Com os códigos já compilados, foi gerado uma janela gráfica com quatro imagens (1-representando a imagem original, 2-representando o negativo da imagem original, 3-representando o resultado de aplicação do filtro passa-baixa, 4--representando o resultado de aplicação do filtro passa-baixa). Janela gráfica mostrada abaixo.



Além de gerar essa janela gráfica, as imagens geradas da aplicação do filtro foram salvas em uma pasta no diretório '/Imagens_com_filtro' para uma melhor comparação com a imagem original.

• Imagem original X imagem com filtro passa-baixa

Analisando as imagens percebe-se que a imagem com o filtro aplicado perde definição e nitidez em relação a imagem original (criando leves borrões nas áreas de contraste). O filtro de média ponderada passa-baixa, como já esperado, acabou diminuindo os contrastes da imagem. Abaixo segue as imagens em alta escala para comparação.

Imagem original

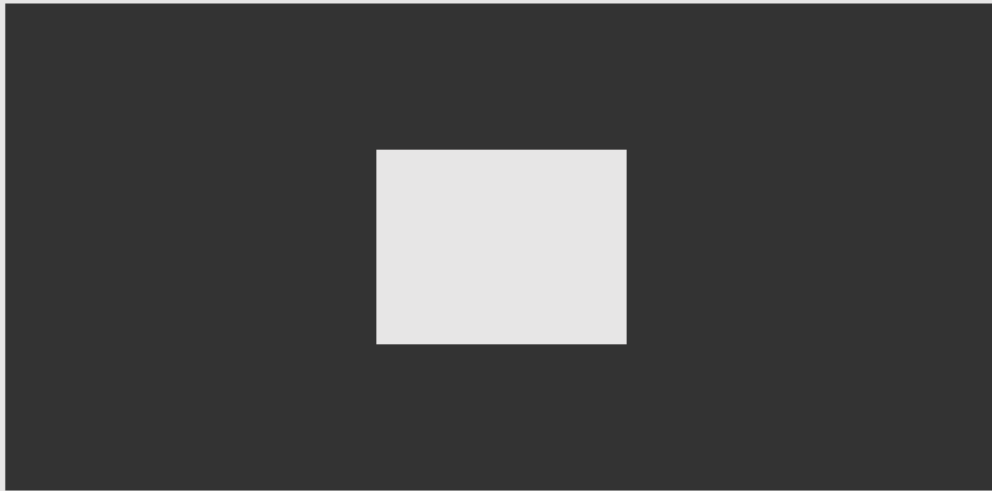
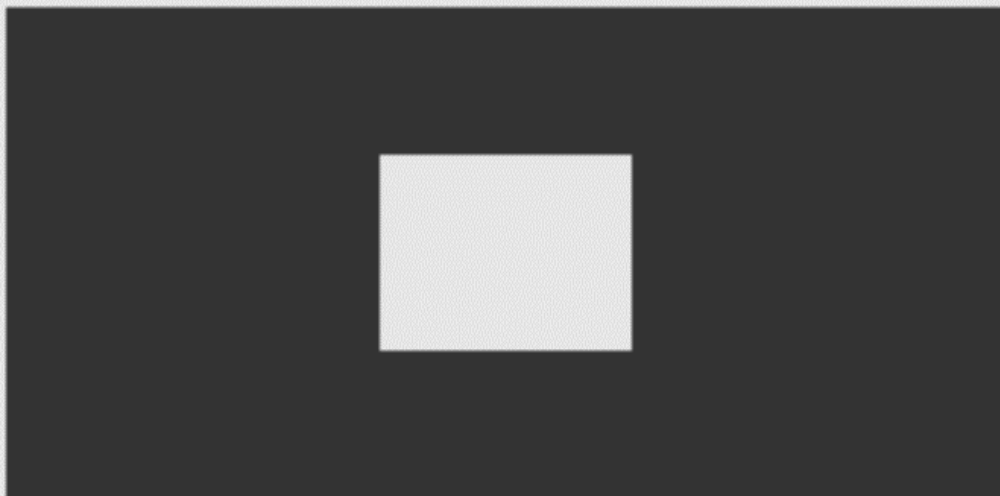


Imagem com filtro aplicado



- **Imagem original X imagem com filtro passa-alta**

Analisando as imagens percebe-se que a imagem com o filtro aplicado só leva em consideração as áreas de alto contraste (descartando todas as áreas de baixo contraste na imagem). O filtro de coeficientes negativos passa-alta, como já esperado, acabou dando mais nitidez nos pontos de contraste. Abaixo segue as imagens em alta escala para comparação.

Imagem original

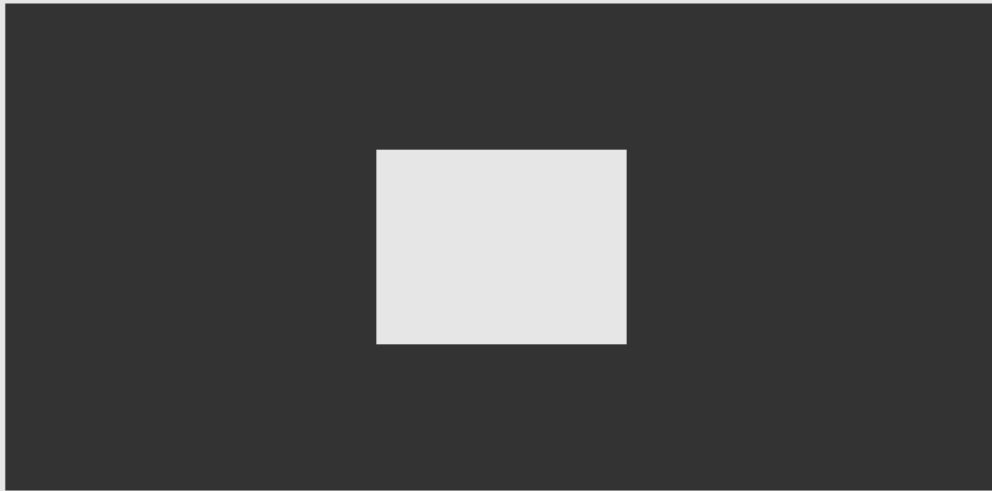
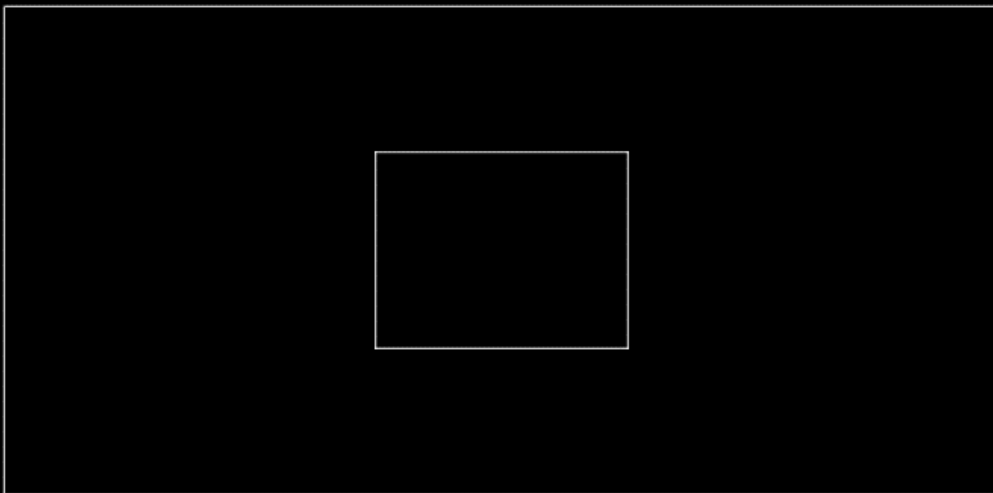


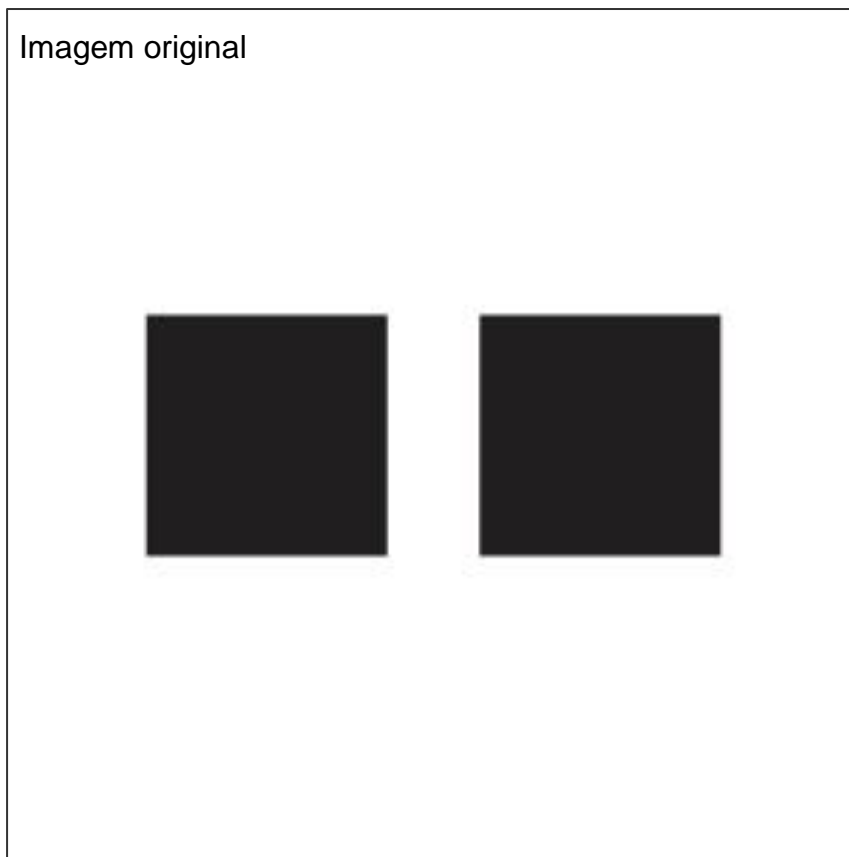
Imagem com filtro aplicado



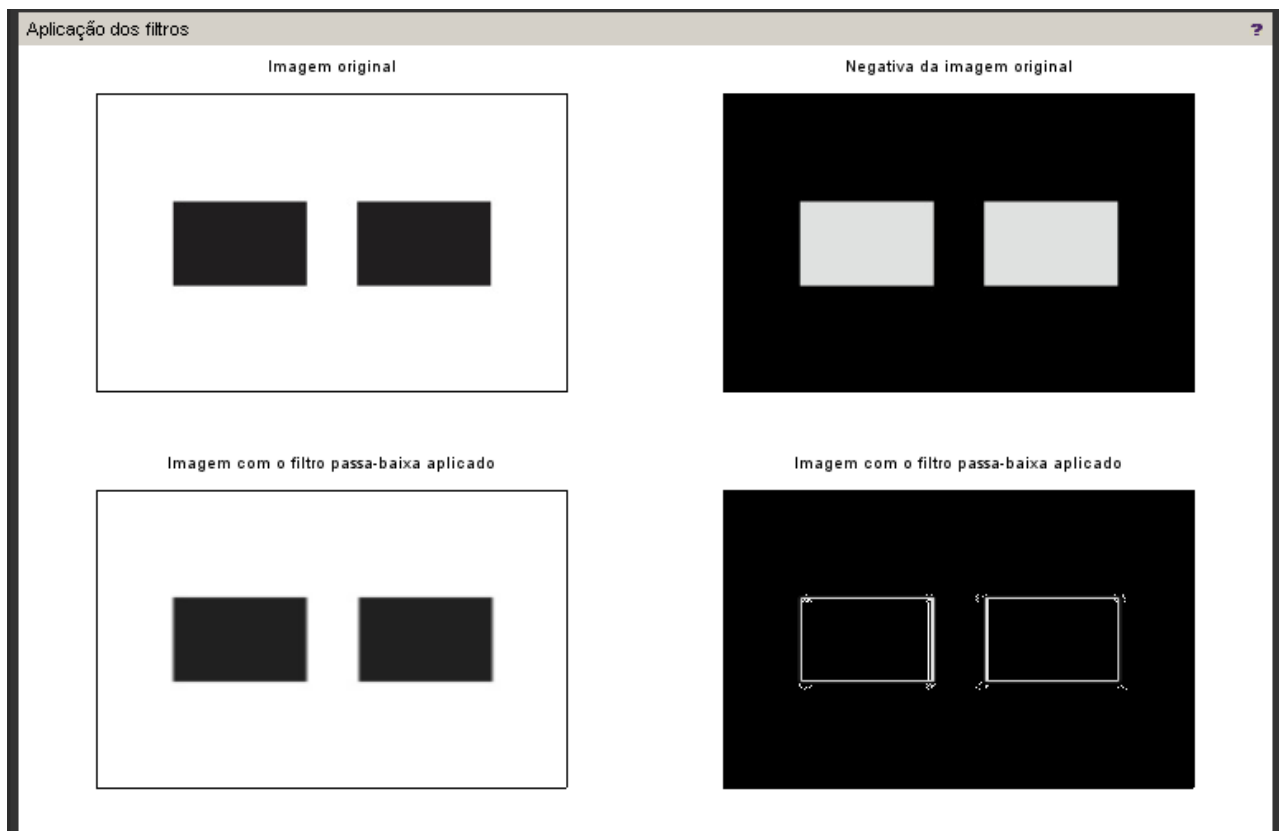
OBS: A imagem original (de própria autoria) foi criada em um formato que não incorpora muitos ruídos, logo, quando aplicado o filtro passa-alta, se observa muita **pouca interferência dos ruídos**.

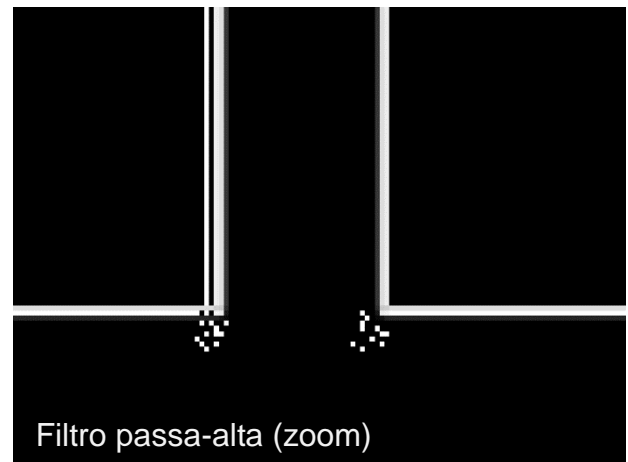
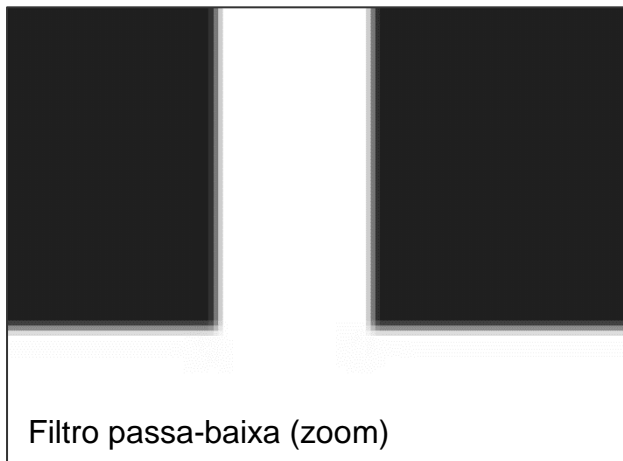
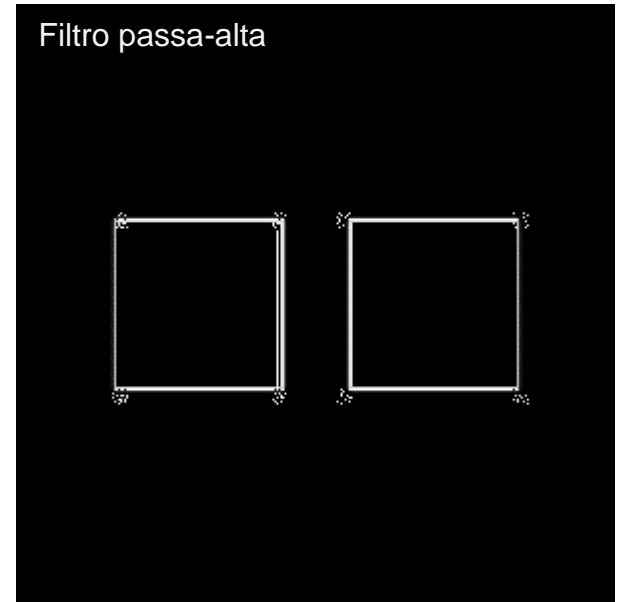
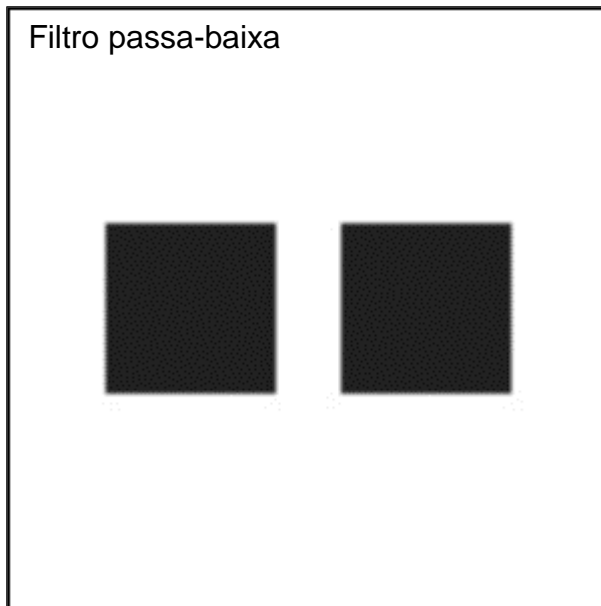
- **Imagem com nível de ruídos considerável**

Foi utilizada uma imagem da internet (com ruídos consideráveis) como imagem base para a aplicação dos filtros. A imagem original segue abaixo.



Compilando o código gerou os seguintes resultados:





Analisando as imagens percebe-se que a imagem com o filtro passa-alta acaba realçando os ruídos da imagem original enquanto o filtro passa-baixa acab amenizando os ruídos.

Compressão de imagens

Introdução

Com o aumento da tecnologia de captura de imagens, a qualidade da imagem também aumenta e assim, também aumentando a memória necessária para armazenar tal imagem.

Tanto para contornar esse problema de aumento de memória (bits necessários para representar a imagem computacionalmente) e garantir uma transferência íntegra da imagem entre meios digitais; A compressão de imagem se baseia na remoção de informações redundantes existentes nas imagens.

A compressão de imagem possui duas categorias de compressão.

- **Não-destrutiva:** Se trata de uma compressão que torna possível reconstruir exatamente a imagem em seu estado original (imagem antes de ter sido efetuada a compressão).
- **Destrutiva:** Se trata de uma compressão que gera perdas nas características das imagens, contudo, permite obter graus de compressão mais elevados (diminuindo a memória necessária para guardar a imagem).

Em relação aos tipos de redundância explorados pelo método de compressão, podemos citar quatro tipos.

- **Codificação (tons, cor, etc):** Ocorre uma redundância nos níveis de cinza ou as cores são codificadas com mais detalhes que o necessário na representação da imagem (possui vários detalhes que podem ser usados para redundância por codificação).
- **Inter-Pixel:** Ocorre uma redundância utilizando relações geométricas ou estruturais entre os pixels, ou seja, a redundância leva em consideração os padrões apresentados nos pixels da imagem.
- **Psicovisuais:** Ocorre uma redundância em detalhes que não vão ser percebidos ou não terão interferência na sensibilidade visual (principalmente devido as limitações do sistema visual humano).
- **Espectral:** Ocorre uma redundância em faixas espectrais específicas, quando os valores espectrais dos pixels na matriz de pixels correspondem a banda espectral especificada para redundância.

LZW (Lempel-ZivWelch)

É uma técnica de compressão não-destrutiva, ou seja, não possui perda de detalhes. A técnica consiste em criar um dicionário de palavras contendo símbolos referentes a codificação de comprimento fixo da imagem (redundância por codificação). O LZW é uma das técnicas de compressão utilizada para criação de arquivos no formato GIF, TIFF e PDF.

A seguir é feito um exemplo retirado de uma bibliografia de referência para demonstrar melhor o funcionamento da técnica.

Inicialmente tem-se a imagem representada na matriz a seguir (com 9 bits por símbolo).

39	39	126	126
39	39	126	126
39	39	126	126
39	39	126	126

Depois foi montado uma tabela de endereços no dicionário de acordo com a entrada.

Posição do Dicionário (Endereço):	Entrada:
0	0
1	1
...	...
255	255
256	—
...	—
511	—

Já sabendo o endereço, agora é possível montar o dicionário de codificação das entradas, tornando possível o reconhecimento da sequência de pixels.

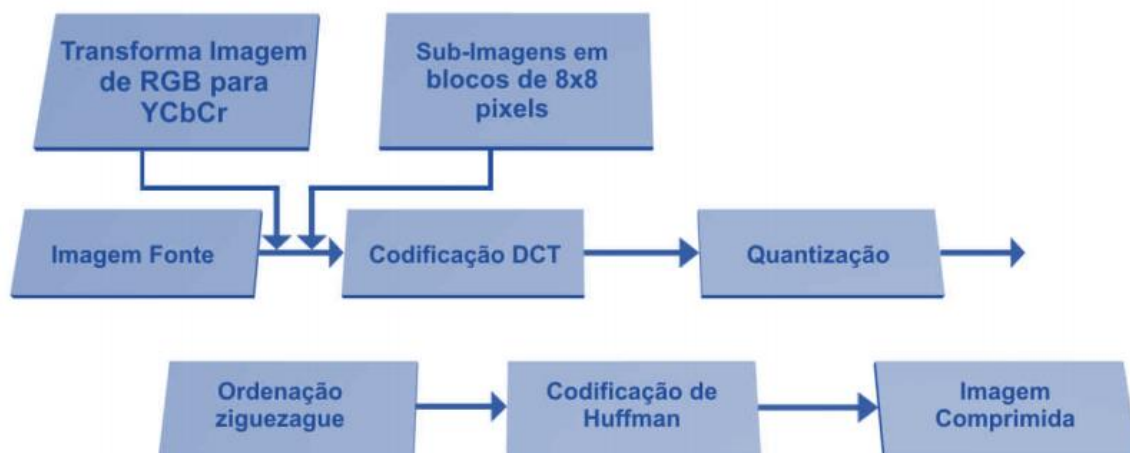
Sequência reconhecida:	<i>Pixel</i> processado:	Saída codificada:	Endereço do Dicionário:	Entrada do Dicionário
	39			
39	39	39	256	39-39
39	126	39	257	39-126
126	126	126	258	126-126
126	39	126	259	126-39
39	39			
39-39	126	256	260	39-39-126
126	126			
126-126	39	258	261	126-126-39
39	39			
39-39	126			
39-39-126	126	260	262	39-39-126-126
126	39			
126-39	39	259	263	126-39-39
39	126			
39-126	126	257	264	39-126-126
126		126		

Compactação JPEG (transformada DCT+Huffman)

A compactação JPEG é uma das mais utilizadas hoje em dia para comprimir e enviar imagens. A compactação JPEG é um processo de compressão de imagens com perdas, contudo, é um dos processos que possui maior nível compressão da imagem (deixando-a bastante compacta).

O processo de compactação possui três passos principais:

- **Computação:** Primeiramente a imagem é transformada para níveis de cinza, e depois, é dividida em blocos de 8x8 pixels e em cada uma desses blocos é calculada a transformada **DCT (Discrete Cossine Transform** - uma técnica de compressão de imagem sem perdas que usa redundância Inter-Pixel, através da transformada discreta de cossenos-).
- **Quantificação:** Os dados gerados pela aplicação do DCT são quantificados através de um **coeficiente de perda**, o qual irá permitir a determinação da proporção de qualidade e tamanho da imagem (diminuindo os valores de alta frequência para minimizar os detalhes). Esse coeficiente de perda percorre os blocos 8x8 utilizando a codificação **RLE** (maneira inteligente de se passar pelos dados - ziguezague - para atingir o máximo de valores nulos na matriz).
- **Codificação:** Nessa última etapa é aplicada a técnica de compressão de **Huffman** (técnica de compressão de imagem sem perdas que consiste em uma redundância codificada na probabilidade de aparecimento de símbolos já mapeados da imagem - organização dos pixels -).



Exemplos Práticos de compressão (Scilab)

Para a implementação de um método de compressão no Scilab também foi utilizado o modulo de processamento de imagem e visão computacional **IPCV** utilizado na prática dos filtros.

Com o intuito de ampliar o experimento e facilitar a aplicação foi utilizado uma técnica geral de compressão e não uma técnica específica (como as apresentadas na introdução).

Foi utilizado O **SVD** (Singular Value Decomposition). Se trata de um procedimento que decompõe a imagem em valores singulares (de simples entendimento e análise). Através da função **SVD** já dada pelo Scilab é possível desmembrar a figura e trabalhar calculando os **erros espectrais** (erro que quantifica a diferença espectral da frequência em relação a frequência original da imagem) e determinando o **número da dimensão dos valores singulares** (medidas das resoluções base de decomposição - pode ser tanto das colunas quanto das linhas das matrizes de decomposição).

Inicialmente a imagem original foi lida pelo sistema e depois aplicada a função **SVD** sobre ela para que futuramente seja possível pegar as características importantes da imagem na forma de matrizes (valores singulares). Código descrito abaixo.

```
163 A = im2double(imread("C:\Users\Arma-X\Desktop\Muitimidia\Trabalho-5\Imagens_originais\bliss(1
eve2).tif"));
164
165 [U,S,V,rk] = svd(A);
166 Valores_singulares = diag(S);
```

Depois foram definidos os parâmetros de testes (Valores singulares da dimensão e percentagens da maior dimensão da imagem) e os parâmetros das matrizes para o processo (linhas e colunas). Também foi incluído o numero total de plots (que simplesmente vai gerando resultados enquanto ainda tem testes para fazer).

```
169 Dimensoes_singulares_para_teste = [5, 15, 25, 35, 45];
170
171 Porcentagem_da_maior_dimensao_para_teste = [1, 0.4, 0.09];
172
173 Total_de_colunas = 3;
174
175 Numero_total_de_plots = 1 + length(Dimensoes_singulares_para_teste) + length(Porcentagem_da_m
aior_dimensao_para_teste);
176
177 Total_de_linhas = (Numero_total_de_plots) / (Total_de_colunas);
178
179 if (modulo(Numero_total_de_plots, Total_de_colunas) <> 0) then
180 ... Total_de_linhas = Total_de_linhas + 1;
181 end
```

Para a primeiro plot é determinado a geração da imagem principal juntamente com a dimensão da matriz original da imagem (dada como sua dimensão singular - dimensão das linhas da matriz $r \times k$)

```
183 // Imagem original
184
185 f = scf(1);
186 clf(1);
187 f.figure_name = 'Compressão utilizando SVD';
188 subplot(Total_de_linhas, Total_de_colunas, 1);
189 imshow(A)
190 xtitle(['Imagem Original'; string(rk) + ' dimensões singulares'])
```

Para os seguintes plots foram geradas as matrizes das imagens comprimidas revendo em consideração as dimensões dos valores singulares para teste. Cada passo do for calcula uma matriz comprimida e plota a imagem respectiva dessa matriz, juntamente com o numero de erro espectral em relação a imagem original. Abaixo segue o código referente a esse processo prático.

```
193 // Dimensões singulares
194
195 for i = 1:length(Dimensoes_singulares_para_teste)
196     Dimensao_singular = Dimensoes_singulares_para_teste(i);
197     Novo_A = U(:, 1:Dimensao_singular) * S(1:Dimensao_singular, 1:Dimensao_singular) * V(:, 1:Dimensao_singular)';
198     Erro = Valores_singulares(Dimensao_singular+1);
199     subplot(Total_de_linhas, Total_de_colunas, i+1);
200     imshow(Novo_A);
201     xtitle([string(Dimensao_singular) + ' dimensões singulares'; ' Erro Espectral Normal = ' + string(Erro)]);
202 end
```

Agora ocorre o mesmo processo, só que agora usando as porcentagens da maior dimensão como teste para gerar matriz comprimida e suas respectivas imagem e também calculando os erros espectrais de cada matriz em cada teste. Abaixo segue o código referente a essa parte.


```

205 // Porcentagem da maior dimensão regular
206
207 for j = 1:length(Porcentagem_da_maior_dimensao_para_teste)
208     Porcentagem_teste = Porcentagem_da_maior_dimensao_para_teste(j);
209     %
210     // Pega os índices dos valores singulares que contribuem mais que X% do maior valor singular
211
212     indicesOfValores_singulares = find(Valores_singulares > (Porcentagem_teste/100)*max(Valores_singulares));
213     Novo_A = U(:, indicesOfValores_singulares)*S(indicesOfValores_singulares, indicesOfValores_singulares)*V(:, indicesOfValores_singulares)';
214     Dimensao_singular = length(indicesOfValores_singulares);
215     Erro = Valores_singulares(Dimensao_singular+1);
216     subplot(Total_de_linhas, Total_de_colunas, 1+length(Dimensoes_singulares_para_teste)+j);
217     imshow(Novo_A);
218     xtitle([string(Dimensao_singular) + ' dimensões singulares'; string(Porcentagem_teste) + ' % do maior valor singular'; ' Erro Espectral Normal = ' + string(Erro)]);
219 end

```

Com todo o código já implementado foi possível gerar uma janela gráfica com vários plots referentes a cada compressão e também foi possível salvar as imagens modificadas em pasta separadas para facilitar comparação com a original. A seguir segue o código extra adicionado em cada for para salvar a imagem na pasta.

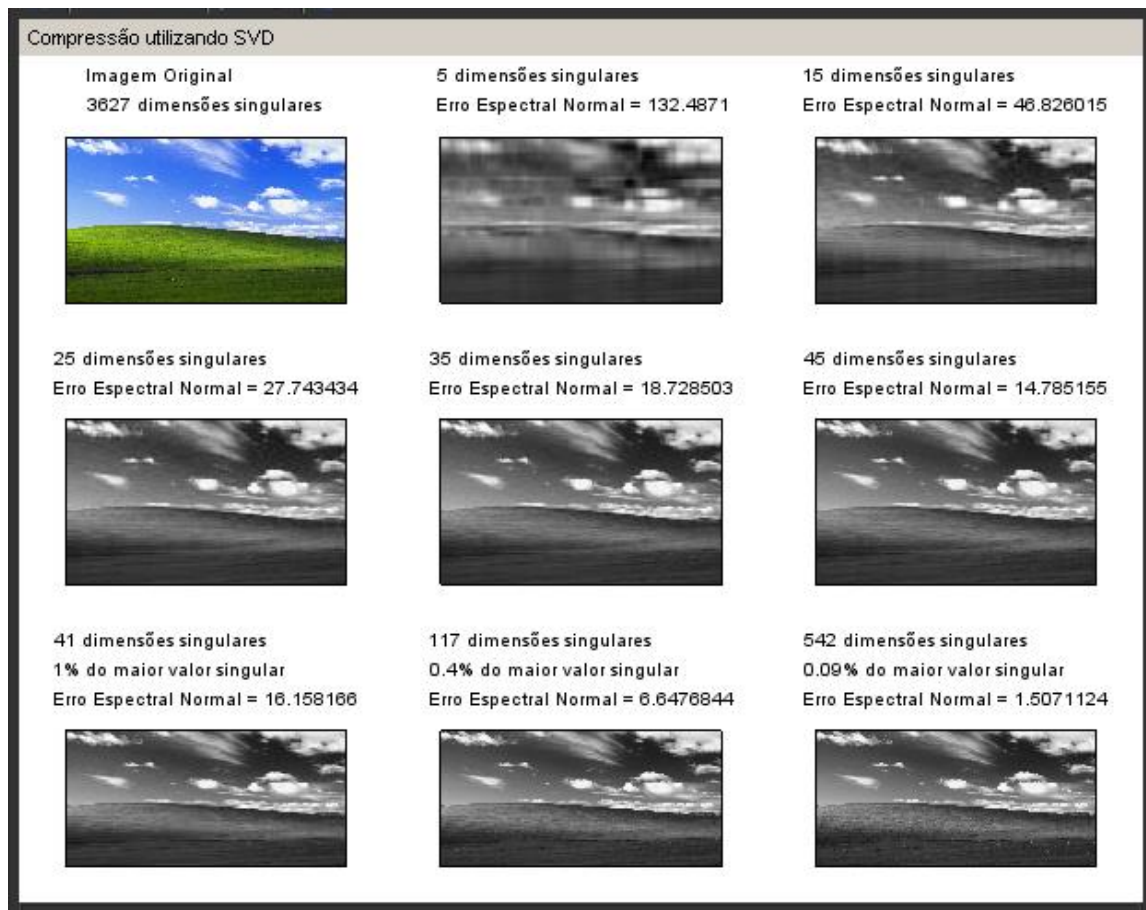
```

203 imwrite(Novo_A, 'C:\Users\Arma-X\Desktop\Muitimidia\Trabalho-5\Imagens_com_compressão\bliss(modl).tif');

```

Resultados:

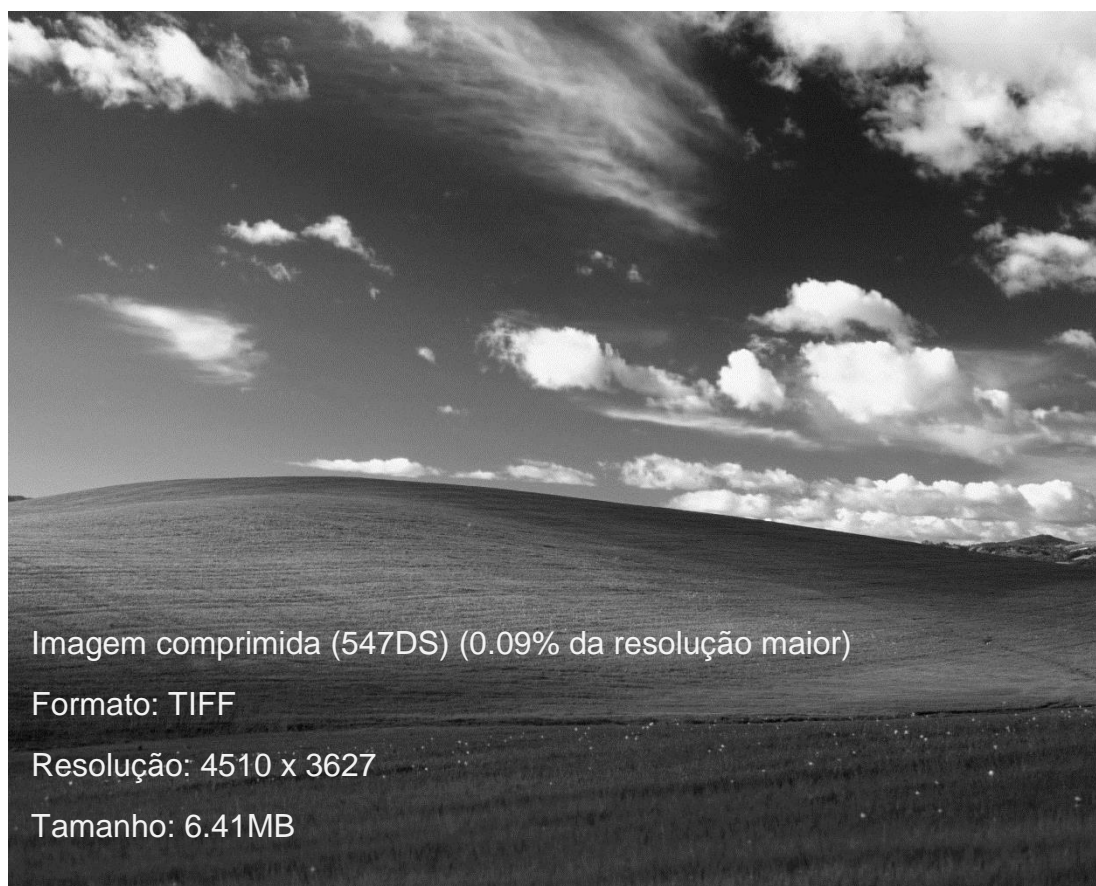
Compilando o código foi gerada a seguinte janela gráfica:



A seguir para melhor compreensão foi mostrada cada figura separadamente. Facilitando a análise.







Referencias

<https://atoms.scilab.org/toolboxes/IPC/1.0>

<https://fileexchange.scilab.org/toolboxes/157000>

<http://computacaografica.ic.uff.br/transparenciasvol2cap8.pdf>

<http://computacaografica.ic.uff.br/transparenciasvol2cap5.pdf>

<http://gec.di.uminho.pt/lesi/vpc0405/Aula06Compress%C3%A3o.pdf>

<https://abertoatedemadrugada.com/2015/05/como-funciona-compressao-jpeg.html>

<https://br.ccm.net/contents/730-compactacao-jpeg>

<http://www2.ic.uff.br/~aconci/CosenosTransformada.pdf>

<http://www.lcs.poli.usp.br/~gstolfi/PPT/APTV0616.pdf>

<https://www.unibalsas.edu.br/wp-content/uploads/2017/01/Artigo-Alisson.pdf>

http://hpc.ct.utfpr.edu.br/~charlie/docs//PID/PID_AULA_08.pdf

<http://professor.pucgoias.edu.br/SiteDocente/admin/arquivosUpload/17303/material/Lab.%205%20-%20Filtros%20passa-baixa%20e%20passa-alta.pdf>

<https://gist.github.com/alek5k/cc299ccf28d71dc9754dc19d40bd541a#file-svdcompression-sce>

https://www.youtube.com/playlist?list=PLA1Sn1TGR3wX_yoUCr5hai4TV9jY2F70V